UNITED STATES PATENT APPLICATION

METHOD AND APPARATUS FOR GENERATING A DISTORTIONLESS PULSE WIDTH MODULATED WAVEFORM

INVENTORS

TUSHAR PRAKASH RINGE
SOURAV ROY

Prakash Nama
Reg. No. 44,255
Schwegman, Lundberg, Woessner, & Kluth, P.A.
1600 TCF Tower
121 South Eighth Street
Minneapolis, Minnesota 55402

ATTORNEY DOCKET NO. 1738.003US1
Client Ref. No. APD2383-1-US

METHOD AND APPARATUS FOR GENERATING A DISTORTIONLESS PULSE WIDTH MODULATED WAVEFORM

Technical Field of the Invention

The present invention relates generally to generating a pulse signal, and more particularly to generating a pulse width modulated (PWM) waveform.

Background of the Invention

Pulse width modulated (PWM) signal generators are used in many kinds of circuits, including embedded controllers, audio processors, and motor control circuits. In general, a PWM signal generator employs a phase-locked loop circuit that generates a rectangular clock signal from a system clock signal and a PWM controller for producing a PWM signal from the rectangular clock signal and a modulating signal. In general the PWM controller is peripheral to a processor and can module the PWM signal by changing period and width values from a user program.

In the case where both period and width values of a PWM signal are to be controlled, it is necessary for the processor to manage the timing of the write of both the period and width values so as to prevent the PWM signal from being generated at undesired times and combinations of period and width values. In addition, changing the period and width values on-the-fly can result in distortion in the form of undesired timing of the PWM waveform and can also result in undesired combinations of period and width values. These drawbacks can make the current PWM waveform generators unsuitable for critical real-time applications.

A type of current PWM waveform generator, which attempts to generate a reliable PWM waveform when both the period and width values are to be controlled, generates a PWM signal depending on a sequence of combinations of period set and width set values which are received from a controller. The above PWM waveform generator utilizes two registers for the period value and another two registers for the width value. A write history flag controls a transfer permit signal to move the period and width values from one set of registers to the next.

One potential problem with the above PWM waveform generator is that it fails to generate the desired waveform when the period and width updates/changes occur on opposite sides of a PWM cycle boundary. Further, the technique does not respond to the updates to provide desired wave forms when the period and width updates continue to occur on opposite sides of the PWM cycle boundary in successive PWM cycles. When generating a PWM waveform, period and/or width updates occur generally in almost every cycle of a PWM signal. If the period and width updates occur on opposite sides of the PWM cycle boundary in each cycle of the PWM signal, then the updates/changes never get implemented by the above-described PWM waveform generator.

It is generally difficult to control the updates to occur at a desired location in a PWM waveform cycle (to occur simultaneously or within a PWM cycle boundary) sequentially when writing to the registers using software running on a processor. It is even more difficult, when using a software program running on a processor, to control the period and width updates to occur at a desired location when the changes have to be made on-the-fly. In addition, adding extra features to the software to monitor and control the timing of the period and width updates increases the load on the processor when the processor is in operation. Therefore, the technique falls short of reliably generating a desired real-time high frequency PWM waveform.

Summary of the Invention

The present invention provides a technique to generate a distortionless, reliable, and predictable real-time PWM waveform based on a sequence of combinations of programmed period and width values received from a processor. In one example embodiment, this is accomplished by storing current period and width values in primary period and width registers, respectively, upon receiving period and width write signals. A secondary period value is then computed using the period and width values. Further, the width value and the secondary period value stored in secondary width and period registers, respectively, upon receiving the width write signal. A down-counter is then with the width value and then the secondary period value is stored in a tertiary period register. The width value is then counted down in

each clock cycle until the width value reaches zero and an expired width signal is outputted and then the secondary period value is loaded. The secondary period value is then counted down in each clock cycle until it reaches zero and an expired period signal is outputted. A PWM signal is then generated based on the expired period and width signals. The process described-above repeats itself for a next set of period and width values received from the processor.

Brief Description of the Drawings

FIG. 1 is a block diagram of one example embodiment of a PWM waveform generator.

FIG. 2 is a timing diagram illustrating all timing signals generated using the PWM waveform generator shown in FIG. 1.

FIG. 3 is another embodiment of the timing diagram illustrating all the timing signals generated using the PWM waveform generator, when the period and width updates occur on opposite sides of a PWM cycle boundary, shown in FIG. 1.

FIG. 4 is another embodiment of the timing diagram illustrating all the timing signals generated using the PWM waveform generator, when generating a lowest possible PWM cycle period, shown in FIG. 1.

FIG. 5 is a flowchart illustrating a method of generating the distortionless PWM waveform on a real-time basis.

Detailed Description of the Invention

The present subject matter provides an architecture to generate a distortionless and predictable real-time PWM waveform based on a sequence of combinations of programmed period and width values received from a processor. In one example embodiment, the PWM waveform is generated using a programmable pulse width modulated (PWM) waveform generator to generate a distortionless and reliable PWM signal having desired combination of width and period. The PWM generator is peripheral to a digital signal processor (DSP), which can change period and width values on-the-fly. This PWM waveform generator is suitable for critical real-time applications where the DSP needs to generate a predictable PWM

waveform. This is accomplished by using two sets of registers to store period and width values and a special tertiary period register for the period value with its timing control. This architecture can also generate a PWM waveform having a width of one clock cycle and a period of two clock cycles on a real-time basis without any distortion. All of the above features are achieved with a minimal increase in hardware over the current techniques.

In the following detailed description of the embodiments of the invention, reference is made to the accompanying drawings that form a part hereof, and in which are shown by way of illustration specific embodiments in which the invention may be practiced. These embodiments are described in sufficient detail to enable those skilled in the art to practice the invention, and it is to be understood that other embodiments may be utilized and that changes may be made without departing from the scope of the present invention. The following detailed description is, therefore, not to be taken in a limiting sense, and the scope of the present invention is defined only by the appended claims.

The terms "cycle" and "period" are used interchangeably throughout the document. In addition, the terms "width" and "duty" are used interchangeably throughout the document. Also, the terms "storage element" and "register" mean the same and refer to a memory element. In addition, the terms "write" and "update" are used interchangeably throughout the document. Further, the term "clock cycle" means system clock cycle. The term "processor" means "digital signal processor", "microcontroller", "audio processor", "microprocessor", and so on.

Referring now to FIG. 1, there is illustrated an example embodiment of a PWM waveform generator 100 according to the present subject matter. The PWM waveform generator 100 includes a primary period register 105 and a primary width register 110. In some embodiments, the primary period register 105 and the primary width register 110 are memory-mapped registers. The PWM waveform generator 100 further includes a subtractor 115 coupled to the primary period register 105 and the primary width register 110. The PWM waveform generator 100 also includes a secondary period register 120 coupled to the subtractor 115 and a secondary width register 125 coupled to the primary width register 110.

As shown in FIG. 1, the PWM waveform generator 100 further includes a tertiary period register 130, a down-counter 135, a state machine 140, a timing controller 145, a configuration register 150, and a final output waveform generator 155. Also as shown in FIG.1, the PWM waveform generator 100 further includes a DSP core 160 operatively coupled to receive program instructions from a user program 165. Further as shown in FIG. 1, the DSP core 160 is coupled to the PWM waveform generator 100 via a data bus 170. Furthermore as shown in FIG. 1, the PWM waveform generator 100 and the DSP core 160 are components of a DSP system-on-chip (SoC).

In operation, in one example embodiment, when a user program 165 executing on the DSP core 160 changes period and width values in the PWM waveform generator 100, the current period and width values are stored in the primary period and width registers 105 and 110, respectively, upon receiving a primary period register write signal and a primary width register write signal, respectively, from the user program 165. In some embodiments, the primary period and width registers 105 and 110 receive the current period and width values and the associated primary register write signals via the data bus 170.

The subtractor 115 then receives the width value from the primary width register 110 and the period value from the primary period register 105 and computes a secondary period value using the received width and period values.

The down-counter 135 counts down the loaded programmed width value with each clock cycle until it reaches zero and generates an EXPIRE signal when the down-counter reaches zero. The state machine 140 generates a state value based on the EXPIRE signal. (The generation of the EXPIRE signal and the state value is explained in more detail with reference to FIG 2). In some embodiments, the state machine 140 receives the EXPIRE signal from the down-counter 135 and outputs the state value. The EXPIRE signal along with the state value is used to generate expired width and expired period signals.

The timing controller 145 then generates a secondary storage element write control signal upon receiving the primary width register write signal from the user program 165. The timing controller 145 further generates a tertiary period register

write signal based on the EXPIRE signal received from the down-counter 135 and the state value from the state machine 140. The tertiary period register write signal is a single clock-cycle pulse at a current PWM cycle boundary. The EXPIRE signal along with the state value is used internally by the timing controller 145 and the final output waveform generator 155 to produce expired width and expired period signals. In addition, the timing controller 145 further generates the width update counter signal based on the expired period signal and a period update counter signal based on the expired width signal. In these embodiments, the width update counter signal and the tertiary period register write signal are same signals in the temporal domain.

The secondary period and width registers 120 and 125 store the secondary period and width values, respectively, upon receiving the secondary storage element write control signal from timing controller 145. The tertiary period register 130 receives the secondary period value from the secondary period register 120 upon receiving the tertiary period register write signal and stores the secondary period value in the tertiary period register 130.

The down-counter 135 then receives the width value from the secondary width register 125 upon receiving the width update counter signal from the timing controller 145 and generates an EXPIRE signal based on the width value. This EXPIRE signal corresponds to the expired width signal. The down-counter 135 to further receive the secondary period value from the tertiary period register 130 upon generating the EXPIRE signal, and again generates the EXPIRE signal based on the period value. This EXPIRE signal corresponds to the expired period signal. In some embodiments, the down-counter 135 counts down the loaded programmed width value with each clock cycle until the down-counter reaches zero and then generates the expired width signal. Also in these embodiments, the down-counter 135 counts down the loaded programmed secondary period value with each clock cycle until the down-counter reaches zero and then generates the EXPIRE signal.

The final output waveform generator 155 then receives the EXPIRE signal from the down-counter 135 and the state value from the state machine 140 to internally generate the expired period and width signals. The final output waveform

generator 155 then generates a PWM signal on a real-time basis as a function of expired period and width signals. The configuration register 150 receives a control signal from the user program 165 and outputs a configuration signal. The final output waveform generator 155 generates the PWM signal or its inverse based on the configuration signal. In these embodiments, the final output waveform generator 155 outputs a PWM signal based on the expired period and width signals and the configuration signal. Also in these embodiments, the final output waveform generator 155 generates a PWM waveform having a width of 1 clock cycle and a period of 2 clock cycles. In some embodiments, the final output waveform generator 155 outputs the PWM signal based on the expired period signal, the expired width signal, the state value, and the configuration signal.

Referring now to FIG. 2, there is illustrated all the timing signals 200 generated, using the PWM waveform generator shown in FIG.1, according to an embodiment of the present invention. Dataflow through the primary period and width registers 105 and 110, the secondary period and width registers 120 and 125, and the tertiary period register 130 are controlled by the timing signals based on the time when the period and width values are updated by the user and also based on the state of the state-machine 140. The down-counter 135 operates on the secondary width value and the tertiary period value. The final output waveform generator 155 either outputs a logic one or logic high, depending on the state. The state of the state machine 140 changes from *state A* to *state B* and vice-versa, as shown in timing signal 260 in FIG. 2, when the down-counter 135 reaches zero. It can be envisioned that an up-counter, in place of the down-counter, is used when using inverted values of period and width, though they are conceptually same.

FIG. 2 further shows the generated output PWM waveform 205. The primary period and width write signals are generated as indicated in timing signals P1 and W1 210 and 215. In some embodiments, the user updates the period value first as indicated in the timing signal 210. Once the user updates the period value, the width value must also be updated as shown in the timing signals 210 and 215. The PWM waveform generator 100 can update only the width, which may be necessary in many cases, such as communication related PWM, where the period

value corresponds to a constant sampling period. In these embodiments, updating the width value is what actually transfers the period and width values to their respective secondary period and width registers 120 and 125 from the primary period and width registers 105 and 110, respectively, on a next clock cycle as shown in timing signals 230, 235, 220, and 225. The transfer of the period and width values from the primary registers to the secondary registers are based on receiving the secondary storage element write control signal as illustrated using timing signal P2W2 240. The secondary period register actually gets the computed secondary period value (period value - width value).

In a PWM cycle, initially a logic high is generated for a specific number of clock cycles corresponding to the width value. Subsequent to generating the logic high, a logic low is generated for a period of secondary period value clock cycles. The state machine 140 is in *state A* in the former case and in *state B* in the latter case as shown in the timing signal 260. A control bit, when set can generate a configuration signal, which in turn can generate a logic low corresponding to *state A* and a logic high corresponding to *state B*. At the beginning of *state A* the tertiary period register 130 is loaded with the secondary period register value as shown in timing signal 255. This corresponds to timing signal P3 250. The tertiary period register 130 prevents any period change to go to the down-counter 135 during a current PWM cycle. Also, the down-counter 135 loads the secondary width register value corresponding to the width update counter signal LD_WDT 280 and starts counting down until the down-counter reaches a value of zero. In the last counting cycle, EXPIRE signal 270 is generated as shown in timing signal 280. This corresponds to the expired width signal. At the end of *state A*, *state B* starts and the down-counter 135 is loaded with the tertiary period register value corresponding to the period update counter signal LD_PRD as shown in timing signals 260 and 285. Again, the down-counter 135 counts down until it reaches a value of zero. Now it again generates the EXPIRE signal as shown in the timing signal 290. This corresponds to the expired period signal. Hence, the down-counter 135 loads the width value when the EXPIRE signal is generated in *state B* as shown in timing signals 290 and 295. Similarly, the down-counter 135 loads a value of (period-

width), i.e., the secondary period value when the EXPIRE signal is generated in *state A*. At other times for every clock period the down-counter decrements the count by unity. This ends a complete period of the output PWM waveform. This mechanism permits high frequency PWM waveforms to be generated from a DSP on-the-fly without any distortion. Adding such a reliability feature to the DSP has a minimal impact on the DSP resource and a minimal increase in the DSP hardware. The above process repeats itself for a next PWM waveform generated by the PWM waveform generator 100.

Referring now to FIG. 3, there is illustrated all the timing signals 300, similar to the timing signals shown in FIG.2, generated using the PWM waveform generator 100 shown in FIG.1, according to another embodiment of the present invention. FIG. 3 shows how the present subject matter solves the reliability problem when a sequence of period and width updates/changes, such as PA, PB, and PC 310 and WA, WB, and WC 320 occur on opposite sides of a PWM cycle boundary 330. The generation of the timing signals 300 has been described-above in detail with reference to FIG. 2. The timing signals shown in FIG. 3 clearly illustrate the generation of the PWM waveform without any distortion when the period and width updates/changes 310 and 320 occur across the PWM boundary 310 in a sequence as shown in PWM cycles 340, 350, and 360.

Referring now to FIG. 4, there is illustrated all the timing signals 400 generated using the PWM waveform generator shown in FIG.1, according to an embodiment of the present invention. Again, the timing signals depicted in FIG. 4 are similar to the timing signals depicted in FIGS. 2 and 3 and the generation of these timing signals has been described in more detail with reference to FIG. 2. FIG. 4 illustrates the seamless generation of the PWM waveforms, having a width of one clock cycle and a period of two clock cycles, without any distortion.

Referring now to FIG. 5, there is illustrated an embodiment of a method 500 according to the present invention. At 510, the method 500 in this example embodiment

receives a sequence of combinations of programmed period and width values and their associated primary period and width write signals from a user program through

a DSP core via a data bus. At 520, each of the received sequence of combinations of programmed period and width values are stored in primary period and width registers.

At 530, the programmed width value is stored by obtaining the programmed width value from the primary width register into a secondary width register upon receiving a secondary storage element write signal from a timing controller.

At 540, a secondary period value is computed by subtracting the programmed width value from the programmed period value. At 550, the computed secondary period value is stored into a secondary period register upon receiving a secondary storage element write control signal from the timing controller. In some embodiments, the secondary storage element write control signal is generated upon receiving the primary width register write signal. At 560, the secondary period value is further stored in a tertiary period register upon receiving a tertiary period register write signal from the timing controller.

Also at 560, a down-counter is loaded with the programmed width value by obtaining the programmed width value from the secondary width register upon receiving the width update counter write signal from the timing controller. The width update counter signal and the tertiary period register write signal are the same in the temporal domain. In some embodiments, the tertiary period register write signal is generated based on a state signal and an EXPIRE signal received from the state machine and the down-counter, respectively. At 570, the loaded programmed width value is counted down with each clock cycle and an expired width signal is produced when the down-counter reaches zero. In addition, a count width signal is generated until the expired width signal is produced.

At 580, the secondary period value is loaded into the down-counter upon producing the expired width signal and upon receiving a period update counter signal from the timing controller. At 590, the loaded secondary period value is counted down until the down-counter reaches zero and an expired period signal is produced. In addition, a count period signal is generated until the expired period signal is produced. In these embodiments, the state signal along with the EXPIRE signal is used to generate the expired period and width signals associated with the

current PWM signal. The state signal includes boundary information of the current PWM signal. Also in these embodiments, the width update and period update counter signals are generated based on the state signal.

At 592, a PWM signal is generated based on the produced expired period and width signals. In some embodiments, a configuration signal is generated such that the configuration signal can generate an inverse of the PWM waveform. In these embodiments, the PWM signal is generated based on the configuration signal and the expired period and width signals. The above method 500 repeats itself for a next programmed period and width values received from the user program through the DSP core.

Although the method 500 includes blocks 510-592 that are arranged serially in the exemplary embodiments, other embodiments of the subject matter may execute two or more blocks in parallel, using multiple processors or a single processor organized two or more virtual machines or sub-processors. Moreover, still other embodiments may implement the blocks as two or more specific interconnected hardware modules with related control and data signals communicated between and through the modules, or as portions of an application-specific integrated circuit. Thus, the exemplary process flow diagrams are applicable to software, firmware, and/or hardware implementations.

The various embodiments of the PWM waveform generator, systems, and methods described herein are applicable generically to generate a distortionless and reliable PWM signal with a desired combination of period and width values. The PWM waveform generator of the present subject matter is peripheral to a digital signal processor (DSP), which can change the period and width values on a real-time basis. The PWM waveform generator described-above is highly suitable for critical real-time applications where the DSP needs to generate distortionless PWM signal. Furthermore, the above-described technique generates high-speed PWM waveforms with the highest attainable frequency of one clock cycle width and two clock cycles in a period. This further enables a highest rate of period and width updates.

The above description is intended to be illustrative, and not restrictive. Many other embodiments will be apparent to those skilled in the art. The scope of the invention should therefore be determined by the appended claims, along with the full scope of equivalents to which such claims are entitled.

5